Analysis of edge detection technique for hardware realization

Nidhi Panda

Department of Electrical Engineering
National Institute of Technology Rourkela

Analysis of edge detection technique for hardware realization

Dissertation submitted to the

National Institute of Technology Rourkela

in partial fulfillment of the requirements

of the degree of

Master of Technology

in

Electrical Engineering

in specialization

Electronic System and Communication

by

Nidhi Panda
(Roll No: 214EE1411)

under the supervision of

Prof. Supratim Gupta

Department of Electrical Engineering,
National Institute of Technology, Rourkela,
Rourkela-769008, Orissa, India

2014-2016

Electrical Engineering
National Institute of Technology Rourkela

Dr. Supratim Gupta
Assistant Professor

May 31, 2016

Supervisor's Certificate

This is to certify that the work presented in this dissertation entitled" Analysis of edge detection technique for hardware realization" by "*Nidhi Panda"*, Roll Number 214EE1411, is a record of original research carried out by her under my supervision and guidance in partial fulfillment of the requirements of the degree of *Master of Technology* in *Electrical Engineering.* Neither this dissertation nor any part of it has been submitted for any degree or diploma to any institute or university in India or abroad.

<Supervisor's Signature>
<Supratim Gupta>

Declaration of Originality

I, ***Nidhi Panda*** , Roll Number ***214EE1411*** hereby declare that this dissertation entitled " ***Analysis of edge detection technique for hardware realization***" represents my original work carried out as a postgraduate student of NIT Rourkela and, to the best of my knowledge, it contains no material previously published or written by another person, nor any material presented for the award of any other degree or diploma of NIT Rourkela or any other institution. Any contribution made to this research by others, with whom I have worked at NIT Rourkela or elsewhere, is explicitly acknowledged in the dissertation. Works of other authors cited in this dissertation have been duly acknowledged under the section "Bibliography". I have also submitted my original research records to the scrutiny committee for evaluation of my dissertation.

May 31, 2016
NIT Rourkela

Nidhi Panda

Dedicated to My Grand Mother.

Acknowledgements

First and foremost, I am truly indebted to my supervisor Professor Supratim Gupta for his inspiration, excellent guidance and unwavering confidence through my study, without which this thesis would not be in its present form. From the bottom of my heart, I would like to express my gratitude to Dr. Supratim Gupta for all the support, patience and encouragement throughout the work.

I thank Mr. Susant Panigrahi and Mr. M.Zefree Lazarus for the stimulating discussions, knowledge sharing sessions, and all the support that they have given me during this work.

I would like to thank my fellow lab mates Kiran Patel, Amrita Maity and Suparna Rooj of Embedded System and Real Time Lab and all my M.Tech and dual degree friends of NIT Rourkela, for their enjoyable and helpful company I had with.

I thank Mr. Sreejith M, former student of ESRT lab NIT, Rourkela from IIT Kharagpur for the help and support that he gave me to learn image processing in VHDL.

My wholehearted gratitude to my parents, Devendra and Madhu Panda, my aunty Sudha Panda and sisters Aparna and Ranu Panda, for their encouragement and support.

Nidhi Panda

Rourkela, May 2016

Abstract

Edge detection plays an important role in image processing and computer vision applications. Different edge detection technique with distinct criteria have been proposed in various literatures. Thus an evaluation of different edge detection techniques is essential to measure their effectiveness over a wide range of natural images with varying applications. Several performance indices for quantitative evaluation of edge detectors may be found in the literature among which Edge Mis-Match error (EMM), F-Measure (FM), Figure of Merit (FOM) and Precision and Recall (PR) curve are most effective. Several experiments on different database containing a wide range of natural and synthetic images illustrate the effectiveness of Canny edge detector over other detectors for varying conditions. Moreover, due to the ever increasing demand for high speed and time critical tasks in many image processing application, we have implemented an efficient hardware architecture for Canny edge detector in VHDL. The studied implementation technique adopts parallel architecture of Field Programmable Gate Array (FPGA) to accelerate the process of edge detection via. Canny's algorithm. In this dissertation, we have simulated the considered architecture in Modelsim 10.4a student edition to demonstrate the potential of parallel processing for edge detection. This analysis and implementation may encourage and serve as a basis building block for several complex computer vision applications. With the advent of Field Programmable Gate Arrays (FPGA), massively parallel architectures

can be developed to accelerate the execution speed of several image processing algorithms. In this work, such a parallel architecture is proposed to accelerate the Canny edge detection algorithm. The architecture is simulated in Modelsim 10.4a student edition platform. This work has wider scope and applications. FPGA based edge detection system can serve as the basic step for implementations of complex computer vision algorithms.

Keywords: *Consensus Ground Truth, Edge Mismatch Error (EMM), F-Measure (FM), Figure of Merit (FOM), FPGA, Precision Recall (PR) curve, VHDL Architecture.*

Contents

List of Abbreviations

Abbreviation	Description
1D	One Dimension
2D	Two Dimension
CE	Common Edges
EMM	Edge Miss Match Error
EO	Excess edges
ET	Excess Thresholded Edge
FN	False negative
FOM	Figure of merit
FP	False positive
FPGA	Field Programmable Gate Array
FPR	False positive rate
PC	Phase Congruency
PR curve	Precision recall curve
TN	True negative
TP	True positive
TPR	True positive rate
VHDL	Very High Speed Integrated Circuit (VHSIC) Hardware Description Language

List Symbols

Symbols	Description
x, y	Spatial coordinates
f	Frequency
d	Dimension
Z	Complex Plane
ϕ	Phase
m	Magnitude
N	Length of Time Series
δ	Distance function
r, θ	Polar Co-ordinates
μ	Mean
σ	Standard Deviation
$I(x, y)$	Image
c	Contrast
$S(x, y)$	Smoothed image
$G(x, y)$	Filter mask

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Edge detection in an image is the fundamental tool in computer vision and image processing for the identification and reconstruction of local properties. In an image most of its shape and structural information is included in edge. So in image identification process, initially edges are detected, while removing less important redundant information. This process aides in retaining objects and shape information of an image with increasing sharpness. However, most of the natural images may be acquired in different conditions with varying properties such as difference in illumination condition, depth discontinuity etc. Thus the process of identifying edges considering a generalized methodology is a challenging task. Most of the edge detector differs in mathematical and algorithmic property. The comparative study of various edge detection techniques is essential to have an understanding on all edge detector. Though several matrices [1] may be found in the literature for comparative analysis, most of these indices measure the performance of edge detectors with respect to reference edge map. In this dissertation, we have considered *consensus* [2] of different edge map to generate ground truth image.

Traditional image processing algorithms are sequential in nature. When

these algorithms are implemented in a real-time system, the response time will be high. In an embedded platform, such algorithms consumes more power because of more number of clock cycles required to execute the algorithm. With the advent of Field Programmable Gate Arrays (FPGA), massively parallel architectures can be developed to accelerate the execution speed of several image processing algorithms. In this work, such a parallel architecture is studied to accelerate the Canny edge detection algorithm. The architecture is simulated in Modelsim 10.4a student edition platform. FPGA based edge detection system can serve as the basic step for implementations of complex computer vision algorithms. Such as in fast face detection in a digital camera, fast object tracking, policing and interactive surveillance and object tracking [3].

1.2 Motivation

1.2.1 Comparative analysis of different edge detection techniques

The estimation of edge in different detectors vary distinctively according to the choice of smoothing filter, the differential operator, and the localization method. Furthermore, few external factors such as: change in contrast, inadequate illumination, image magnification and blurring may affect edge localization and detection in various detectors. Thus an evaluation of different edge detection techniques is essential to measure their effectiveness over a wide range of natural images with varying applications. Several performance indices may be found in the literature for quantitative evaluation of different edge detection methodologies. However, most of the performance indices are evaluated with respect to the ground truth edge map. In image processing applications the reference image generation can be categorized broadly into three approaches.

- Artificial approach

- Real and Manually annotated approach
- Consensus approach

Artificial approaches are simplest. However many natural images contain objects with different textural characteristic, various percentage of homogenous regions and details, that may not be analyzed using this approach. The second approach is real and manually annotated approach, since the reference edge map generated in this approach relies on human observer, it is neither automatic nor efficient (both in terms of accuracy and time complexity). Furthermore, different subject annotate different pixel position to generate several reference edge map for same image. Thus this approach also have limited applications in edge detection performance evaluation. The need for automatic, accurate and time efficient method for reference image generation motivates many researcher in last few decades. The consensus approach is one such algorithm(s) (with different parameter values) used for ground truth edge map generation.

1.2.2 Hardware Image Processing

A microcontroller/dsp processor executes algorithms sequences sequentially. If multiple hardware circuits can be designed to carry out different algorithm sequences in parallel, there will be considerable increase in overall execution speed. Suppose a system has to be designed such that the brightness of the incoming frames has to be increased. Brightness of an image can be increased by multiplying each pixel gray level with a constant α, and then adding a gain constant β to it as in equation

$$g(i,j) = f(i,j) \times \alpha + \beta \tag{1.1}$$

In a typical microcontroller/dsp processor based design, this will involve storing the frames in a buffer, and then performing the operations mentioned

in Eq. 1.1 to each pixel gray level, in a loop. Suppose each addition instruction takes 12 clock cycle, and each multiplication instruction takes 36 clock cycle, then total number of clock cycles required to process one pixel will be 48. If the incoming frames are of size 100×100, then such a design will need $100 \times 100 \times 48$ clock cycles to process the entire frame. Now suppose, there are 10000 adder and multiplier circuits, one cosponsoring to each pixel. In such a design, all the pixels can be processed in parallel. Thus total operation can be implemented in just two clock cycles. In principle, such a system will be 12000 times faster than system based on sequential processor. In actual designs, algorithm will be divided in to parallel blocks and will be executed simultaneously. In a nutshell, the significant increase in processing speed is the major motivation behind hardware image processing. If the processing time is less, the power consumption can also be reduced. Hence it can be observed that the parallel processor based hardware implementation of image processing systems aid better performance in time critical applications. In the current scenario, most of the image processing algorithms are running in a sequential environment. Here, we have investigated on a architecture for FPGA based edge detection technique, which may have grater significance and scope in time critical applications.

1.3 Objectives

The salient objectives of the thesis are:

i. Comparison of various edge detection technique.

   a) Automatic generation of ground truth edge map using consensus of various edge detectors.

   b) Quantitative performance evaluation of considered edge detectors adopting four different measures: EMM, FOM, F-Measure and PR curve.

ii. Hardware architecture development and implementation of best edge detector in FPGA.

1.4 Thesis Organization

The rest of the thesis is organized as follows: Chapter 2 classifies different edge detection techniques and discusses each detection algorithm in detail. Chapter 3 measures the performance of reported edge detection technique with respect to a reference edge map generated from the consensus of each method. The design of VHDL test bench for edge detection is studied in chapter 4. Chapter 5 elaborates about architecture of Canny edge detection algorithm in VHDL. The simulation results for edge detection techniques both in VHDL and MATLAB are presented and discussed in chapter 6. Chapter 7 concludes and provides future scope of this thesis.

Chapter 2

Edge Detection Methodologies

2.1 Introduction

In many image and computer vision applications, it is required to capture the distinct properties of objects in an image. These properties may be characterize by the photometrical, geometrical and physical phenomenon of the object, which can be distinguished in local maxima, discontinuity (step edges) and junctions [4]. The edge detection techniques are adapted to localize and identify these physical phenomena of the image. In ideal situation, the result of edge detector to any image covers all the discontinuities. However, as it has been proven that discontinuities in intensity corresponds to the change in depth, surface orientation, material properties and illumination. It is almost impossible to get to capture and localize these edges, precisely. Thus different type of edge detectors [5] are used for specific applications.

In most of the detectors the process of detecting edges may be divided into the following three steps.Assuming image is corrupted with additive Gaussian noise, in the first step image is smoothen using low pass filter. This step not only reduces the noise but also suppresses the edges. Thus a trade off between smoothing and edge preservation is essential. The second step is a process of finding edges by using high pass filters. Assuming image is corrupted with

additive Gaussian noise, in the first step image is smoothen using low pass filter. This step not only reduces the noise but also suppresses the edges. Thus a trade off between smoothing and edge preservation is essential.

1. Classical method
    - Roberts Operator
    - Sobel Operator
    - Prewitts operator
2. Phase based method
    - Phase Congruency
3. Gaussian based method
    - LOG operator
    - Zero crossing detector
    - Canny's edge detector

2.2 Classical method

Popular edge detection methods which falls in this categories are Roberts [6], Sobel[7], Prewitt[7] and Kritch operator[7]. In this search based approach, initially a first order derivative based mask is used to compute the gradient magnitude to measure edge strength (as shown in Eq.2.2). Then the orientation of edges are estimated using Eq.2.2 for gradient direction. Though this approach is computationally inexpensive, it is highly sensitive to noise in lack of filtering/smoothing.

$$M(i,j) = \sqrt{G_x^2 + G_y^2} \tag{2.1}$$

$$\theta(i,j) = \tan[\frac{G_y}{G_x}] \tag{2.2}$$

Here,

⋆ G_x = Gradient in x-direction

⋆ G_y = Gradient in y-direction

⋆ M(i,j) = Edge strength

⋆ θ = Edge direction

To get an approximation of edges only magnitude will be enough. Therefore we use thresholding to find the edges in the gradient image for classical approach.

2.2.1 Roberts operator

This operator was proposed by Lawrence Roberts in 1963[6]. The esse of Roberts work is to use discrete differentiation to find the gradient of the image. This can be done by using a 2×2 mask as shown in figure 2.1a and 2.1b.

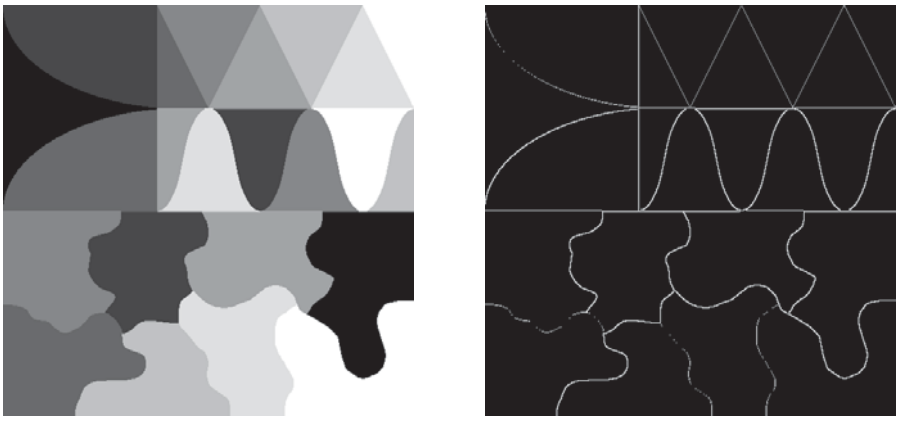
(a) (b)

Figure 2.1: Roberts Mask ,(a) x direction, (b) y direction

As this operator is using 2x2 mask it is computationally very simple but it lacks in efficiency of edge detection.

2.2.2 Sobel operator

It is the most popular edge detector in classical edge detectors. It is proposed by Irwin Sobel & Gary Feldman in 1970 [8]. It uses 3x3 convolution mask in horizontal and vertical direction to find the gradient. The masks as shown in figure 2.3a and 2.3b (known as Sobel's mask) not only computes the gradient



(a) (b)

Figure 2.2: Roberts Mask , (a) input image, (b) edge image

in both X and Y directions but also provides an extra weighting factor of 2, that smoothens the image and reduces noise.

-1	-2	-1
0	0	0
1	2	1

(a)

-1	0	1
-2	0	2
-1	0	1

(b)

Figure 2.3: Sobel Mask, (a) x direction, (b) y direction

(a) (b)

Figure 2.4: Sobel Mask, (a) input image, (b) edge image

This mask is combination of both averaging and differentiation operation,

which makes it more immune to noise than Robert's operator. The advantage of Sobel mask is that, it is very simple and the coefficients values and kernel size can be changed according to requirement. The edges of image after Sobel are thick so they can not be used where the peripheral contour is area of interest.

2.2.3 Prewitt operator

This operator was given by Judith M. S. Prewitt in 1970. It also uses a 3x3 mask with only values 0, 1, & -1. Prewitt Operator compute the edges by using the mask as shown in figure 2.5a and 2.5b. The result of prewitt operator is as shown in figure 2.6b.

-1	-1	-1
0	0	0
1	1	1

(a) Prewitt mask along x and y direction

-1	-1	-1
0	0	0
1	1	1

(b)

Figure 2.5: Prewitt Mask , (a) x direction, (b) y direction

(a) (b)

Figure 2.6: Prewitt Mask , (a) input image, (b) edge image

The result of Prewitt operator is shown in figure 2.6b. As this mask does

not give extra weight to center pixel value it is more sensitive to noise then Sobel mask. Also the mask coefficient and the kernel size are fixed for this type of operator.

2.3 Phase Based method

A clear idea about feature can be extracted at any angle by Congruency of phases. Angle at which congruency occurs suggest the edge type, i.e step or delta. Morrone et al.[9] and Morrone and Owens developed a Local Energy Model. Other work in this modelling of edges can be model of feature perception can be obtained in Morrone and Burr , Owens et al.[10], Venkatesh and Owens , and Kovesi [11]. Morrone and Burr [9] showed that this modelling clearly shows psychophysical effects in edge perception.

The phase congruency measure given by Morrone et al. [9] is

$$PC_1(z) = \frac{|E(z)|}{\sum_n A_n(z)} \tag{2.3}$$

here,

$\star$ $|E(z)|$ = Local energy

$\star$ $A_n(z)$ = Amplitude of Fourier component at location x.

Therefore, phase congruency (PC) can be defined as ratio of $|E(z)|$ to the complete path length used by the local Fourier components till the end point. Here, the ratio of $|E(z)|$ to the $A_n(z)$ will be 1, only if all Fourier components will be phase, i.e all complex vectors will be aligned. And the ratio will be zero if no coherence in phases occur. Phase Congruency Methods are better than methods which are gradient based.As the gradient based methods are easily effected by blurring, illumination condition and magnification. PC is a limitless quantity which does not vary whit environmental condition. This

PC measure is a function of cosine of the deviation of each phase component from the mean value. This can be expressed as shown in Eq. 2.4.

$$PC_1(z) = \frac{\sum_n A_n(\cos(\phi(z) - \overline{\phi}(z)))}{\sum_n A_n(z)} \tag{2.4}$$

This phase congruency measure does not provide good localization and are also not robust to noise. Pater Kovesi [11] developed a modified measure of Phase congruency which provide more localization and also less sensitive to noise but the problem with it is computationally very intensive. This Modified Measure of Phase congruency is as shown in Eq.2.5.

$$PC(z) = \frac{W(z)\lfloor E(z) - T \rfloor}{\sum_n A_n(z) + \epsilon} \tag{2.5}$$

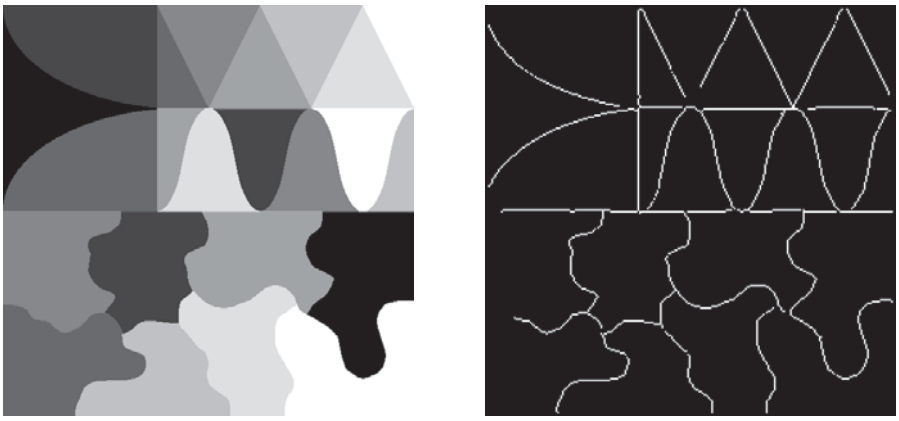
- $\star$ T = Threshold
- $\star$ W(z) = Weighting function
- $\star$ ϵ = Small positive constant
- $\star$ $E(z) = \sqrt{F(z)^2 + H(z)^2}$

This can also be represented as cosine minus the magnitude of the sine of the phase deviation as shown in Eq. 2.6.

$$PC_2(z) = \frac{W(z)\lfloor A_n(z)(\cos(\phi(z) - \overline{\phi(z)})) - |\sin(\phi(z) - \overline{\phi}(z))| - T \rfloor}{\sum_n A_n(z) + \epsilon} \tag{2.6}$$

2.4 Gaussian based method

In many image processing applications Gaussian based methods for feature extraction are highly appreciated and utilized for edge detection. It has been proven that these filters play a major role in the fields of biological vision specifically for the human-vision systems. Gaussian filter-based edge detection techniques are developed on the basis of physiological examinations



(a) (b)

Figure 2.7: Phase congruency, (a) input image, (b) edge image

and prominent properties of the Gaussian function which enables to execute edge detection analysis.

2.4.1 LOG operator

The Gaussian based approach for edge detection was first proposed By Marr and Hildreth [12]. The intensity variation in an image occurs at different level is considered for feature extraction. Thus a smoothing filter at each scale is adopted to serve the purpose. As at different scales a single filter may not be optimum, Marr and Hildreth suggested 2D Gaussian function which can be defined in Eq. 2.7.

$$G(x,y) = \frac{1}{2\pi\sigma^2} \exp^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{2.7}$$

where

$\star$ σ = Standard deviation

$\star$ (x, y) = Pixel's cartesian coordinates

As it is already proven that the Gaussian filters produce different set of image with distinct level of smoothing, a zero crossing of second derivative of Gaussian to find the edges. Such a second order derivative based scalar

approximated filter as in Eq. 2.8 known as Laplacian of Gaussian (LOG) is considered Eq. 2.8 for edge computation.

$$g(x,y) = \nabla^2[G(x,y) * I(x,y)] \tag{2.8}$$

where

⋆ ∇= Laplacian operator

⋆ g(x,y) = Image after Laplacian

⋆ G(x,y) = Smoothed image

⋆ I(x,y)=Input image

LOG is an orientation independent filter. It breaks down at curves, corners and on the locations where intensity value varies nonlinearly along the edges. Thus this operator has limitation in detecting edges at these locations. According to Marr and Hildreth, we can implement smoothing and differentiation just by using the Laplacian of Gaussian function. The LOG with scale σ is as in Eq.2.8.

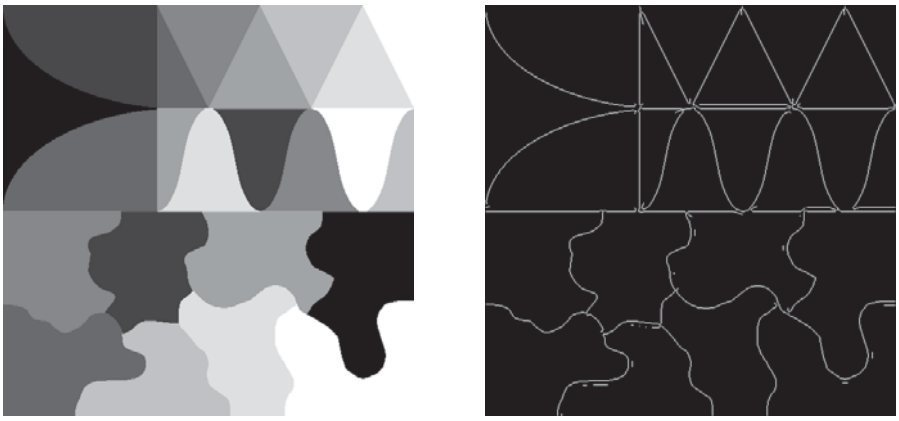
$$f_\sigma(x,y) = \nabla^2 G(x,y) = [\frac{x^2+y^2-2\sigma^2}{\sigma^4}] \exp^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{2.9}$$

The Marr-Hildreth algorithm for edge detection can be wrap-up in following points

- Smoothing of image by using Gaussian filter obtained by Eq. 2.7.
- Calculation of Laplacian on the image by using Eq. 2.8
- Search for zero crossing in image

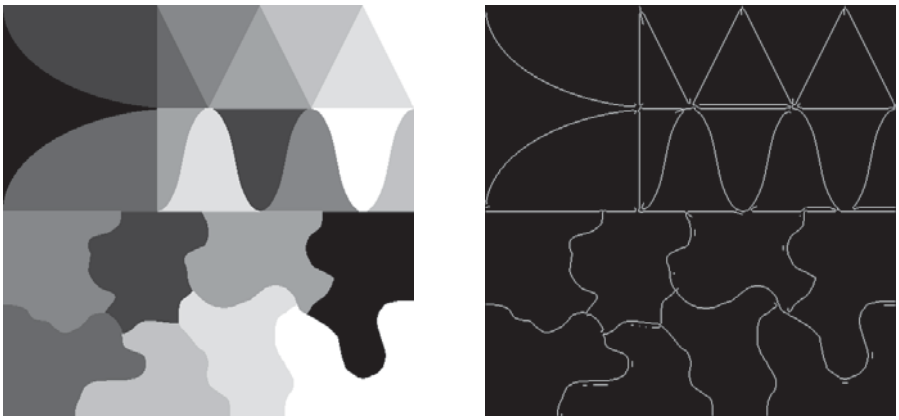
2.4.2 Canny's operator

Canny's approach for edge detection is based on three basic criteria [13].



(a) (b)

Figure 2.8: LOG, (a) input image, (b) edge image



(a) (b)

Figure 2.9: Zero-crossing, (a) input image, (b) edge image

1. **Good detection** - The probability of the detection of the true edge points must be high enough.
2. **Good Localization** - The edge points must be well localized i.e. they must be nearer to the true edge point.
3. **Single edge response**- There must be only one response to the single edge.

The esse of canny's work is to proof these criteria mathematically and finding an optimal solution to it. It is almost impossible to satisfy all these

objectives. Therefore numerical optimization is used with 1D step edges, with additive white Gaussian noise. It is observed that first derivative of Gaussian gives good approximation. But 1D approximation is applied to the direction of the edge normal which is unknown beforehand. Thus the following steps is suggested for better approximation.

- Smoothing - A 2D filtering is used for smoothing and noise reduction.
- Gradient calculation - Any edge detection mask can be used to find the horizontal and vertical gradient.
- Non-Maximum suppression - Suppress the non edge pixel and increases localization.
- Hysteresis thresholding - It removes the false edges by using two threshold value.

An overview of Canny's edge detection framework is shown in Fig.fig:ch2: Canny Edge detector, which utilizes all mentioned steps for feature extraction.

2.5 Smoothing

Smoothing is the first step of Canny edge detector. Image acquisition, limits of digital system and ambient conditions corrupts the image. Generally the noise, observed in images is Gaussian and additive in nature. So smoothing is essential. For smoothing Gaussian mask is used. As literature suggests that volume under the 2D Gaussian surface, between $\pm 3\sigma$ about the mean is 99.7%. Therefore the size of $n \times n$ Gaussian filter should be greater than or equal to 6σ. The 2D Gaussian function is given in Eq.2.11. Gaussian mask suppresses the high frequency components. As noise is high frequency component it minimizes noise but also causes loss of information [5]. In image

processing we stabiles a tradeoff between noise reduction and preservation of edge information.

$$G(x, y) = \exp^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{2.10}$$

If input image is $I(x, y)$ and smoothed image is $S(x, y)$ then, after Gaussian the smoothed image is

$$S(x, y) = G(x, y) * I(x, y) \tag{2.11}$$

Here we use a 3×3 window with sigma equals to 0.5.

0.0113	0.0838	0.0113
0.0838	0.6193	0.0838
0.0113	0.0838	0.0113

(a)

(b)

Figure 2.10: Smoothing operation, (a) Gaussian Mask of 3×3,(b) Smoothed image

2.6 Gradient Calculation

The first order differentiation operator, similar to Sobel's mask in figure 2.3a and 2.3b is used for the computation of gradient in horizontal and vertical direction as formulated in Eq.2.12 and equ:ch4:Gradient Calculation in y. These are used to find the magnitude i.e. edge strength as in Eq. 2.14 and direction as in Eq. 2.15 at every pixel position. Gradient magnitude can be calculated either by Euclidean distance measure by applying Pythagoras theorem or by using Manhattan distance measure. Manhattan distance measure is an approximation method for reducing the computation complexity.

Image gradient provides an estimation about the edges, however as shown in figure 2.13a the edges in the image are not properly localized. Thus we find the gradient direction which can be used in the suppression of local edge points in the non maximum suppression.

$$\frac{\delta S}{\delta x} = S(i,j) * G(i,j) \tag{2.12}$$

$$\frac{\delta S}{\delta y} = S(i,j) * G(i,j) \tag{2.13}$$

$$M(i,j) = \sqrt{\frac{\delta S}{\delta x} + \frac{\delta S}{\delta y}} \tag{2.14}$$

$$\theta(i,j) = \tan\left[\frac{\frac{\delta S}{\delta y}}{\frac{\delta S}{\delta x}}\right] \tag{2.15}$$

(a)

Figure 2.11: Gradient magnitude image

2.7 Non Maxima Suppression

In this step it scan the gradient magnitude image along the edge direction. If the pixel values does not belongs to local maxima we mark it as no edge point. Thus we suppress all edge point which are not the part of local maxima.

Therefore, the process of non maxima suppression is also known as thining process. The steps for the calculation of non maxima suppression is as follows.

1. Round off the gradient direction into 4 direction as shown in figure 2.12.
2. Compare the current pixel with its two neighborhood along the direction.
3. If the magnitude of the current pixel is less than any of the magnitude of its neighbor along the direction replace the current pixel with zero i.e. no edge. Else the current pixel is marked as edge point.

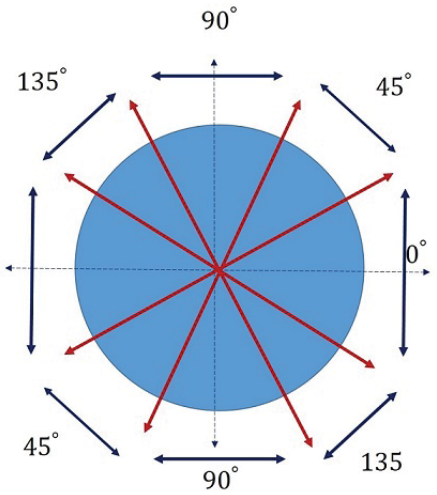
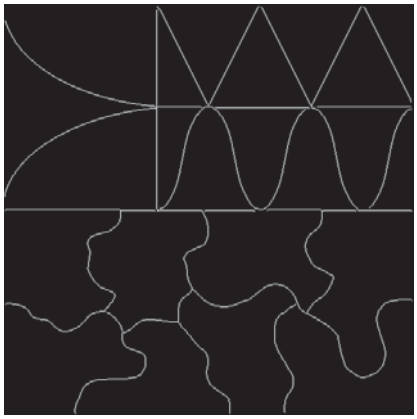


Figure 2.12: The angle approximation of edge normals in four predefined angles

(a)

Figure 2.13: Image after nonmaxima suppression



(a)

Figure 2.14: Edge image after Hysteresis thresholding

2.8 Hysteresis Thresholding

The image we get after non-maxima suppression may have number of false edge which are observed due to the high frequency components of noise. The easiest way to remove this is thresholding. Thresholding is the simplest way to suppress both false and noisy edges. Thus a better localization is achieved by using two thresholds, which excels in following ways:

1. Detects strong edges that are having strength greater than high threshold value.

2. Detects weak edge pixels which are having strength in between high and low threshold value.

3. Suppresses edge pixels which are having edge strength less than lower threshold value. We suppress these pixels.

As Canny's algorithm includes smoothing, non maxima suppression and hysteresis for noise removal, localization and false edge removal respectively. It provides better detection, localization, and single point response. The demerit of this detector is its time complexity.

2.9 Discussion

In this chapter we revisited and discussed on different types of edge detection algorithm in brief. The results thus obtained from all the methods indicate that the performance of different edge detectors vary distinctively. Therefore comparison on different edge detector is essential.

Chapter 3

Performance Evaluation

3.1 Introduction

As discussed in previous Chapter 2, the estimation of edge in different detectors vary distinctively according to the choice of smoothing filter, the differential operator, and the localization method. Furthermore, few external factors such as: change in contrast, inadequate illumination, image magnification and blurring may affect edge localization and detection in various detectors. Thus an evaluation of different edge detection techniques is essential to measure their effectiveness over a wide range of natural images with varying applications. Several performance indices such as: Edge miss match error (EMM)[1], F-measure[1], figure of merit (FOM)[1] and Precision recall (PR) curve [14] may be found in the literature for quantitative evaluation of different edge detection methodologies. However, most of the performance indices are evaluated with respect to the ground truth edge map (structure) [2]. In this chapter along with different quantitative metrics, we will also discuss various methods of reference image generations both via. manually (by visual inspection) and automatically.

3.2 Ground Truth Generation

Quantitative comparison of different edge detection technique needs the use of performance measures. These measure need a reliable ground truth or reference image for evaluation. In image processing applications for edge detection the process of generation of reference image may be classified broadly into three

- Artificial approach
- Real and Manually annotated approach
- Consensus approach

3.2.1 Artificial approach

This is the simplest way of generation of the ground truth image. Artificial approaches are only limited in generating reference image for synthetic image. However many natural images contain objects with different textural characteristic, various percentage of homogenous regions and details, that may not be analyzed using this approach. Thus most of the researchers do not consider results based on synthetic images as convincing and still wish to see results on natural images.

3.2.2 Real and Manually annotated approach

In this approach the natural images are examined manually and edges in the image are marked by the experts in the field. Since the reference edge map generated in this approach relies on human observer, it is neither automatic nor efficient (both in terms of accuracy and time complexity). Furthermore, different subject annotate different pixel position to generate several reference edge map for same image. It is also possible that the same annotator can use different criteria for the generation of different reference images from a single image. Owing to these difficulties the quantitative measure is generally done

by using synthetic images. Thus this approach also have limited applications in edge detection performance evaluation.

3.2.3 Consensus approach

The need for automatic, accurate and time efficient method for reference image generation motivates many researcher in last few decades. The consensus approach is one such algorithm(s) (with different parameter values) used for ground truth edge map generation [2, 15, 16]. These ground truth images are easy to generate and could be used for performance evaluation. Bryant and Bouldin [15] have compared six edge detectors with relative grading method.

They have used True positive (TP) and True negative (TN) statistics along with consensus by using following agreement criterion, is considered for reference edge generation. Any point will be a part of ground truth if minimum K number of edge detector will detect it. Where $1 \leq K \leq N$, if N is total number of edge detector. However, the choice of K is biased and it penalizes those algorithms, that didn't agree with other algorithms [16]. To overcome this [2] proposed a method in which the selection of K is optimized and automated. In this approach each detector each proposed method search for all levels of consensus and the most appropriate is selected. Thus, the effect of the failings of the some algorithms does not affect with the higher level of consensus selection. In the earlier method Yitzhaky and Peli [17] proposed an evaluation technique of edge detectors which uses a statistical objective performance analysis and the detector's parameter selection by using consensus method. This algorithm may be used to compare output of different edge detector binary format. [2] have also considered the same assumption for reference image creation. For the comparison of different edge detectors it is not necessary to have an accurate, reliable and automatic algorithm for reference binary edge map generation. This framework, initially uses different edge detectors to generate various consensus ground truth images using

a voting technique [2]. Then the optimum consensus level K is selected using minimean and minimax approach from various Baddeley measures [18]. Here, we have considered all the discussed edge detectors in Chapter 2 for automatic reference image generation using consensus method. The block diagram for this framework is shown in figure 3.1 and each step of this algorithm is discussed as follows:

1. N different edge detectors are applied.on any natural image to find N different edge maps (outputs O_j) as $O_j, j\varepsilon\{1, 2, \ldots, N\}$.
2. These N output images are used to generate different consensus images C_k such as C_k is consisting of edge pixel which as defined as edge point in at least j output edge images.
3. These consensus images are compared with the each output image O_j , to find the empirical discrepancy measure between consensus and edge image.

$$Discrepancy\ measure = V_{jk} = D(O_j, C_k) \tag{3.1}$$

4. The discrepancy measure corresponding to each consensus image are processed using merging method to find its global consensus value G_K.
5. Finally an optimization operation is done to compute the optimum consensus level that optimize the N consensus values.

The optimum vote or consensus level is chosen for reference image using two novel approaches namely: Minimean and Minimax [2].

3.2.4 Minimean Method

The optimum consensus level can be determined by taking the minimum value of the mean of k^{th} where $k\varepsilon(1 \ldots N)$ level consensus and output of each of the methods. It can be expressed as follows,

$$G_k = \frac{1}{N}\sum_{i=1}^{N}\{V_{j,k} \tag{3.2}$$

$$k_{optimum}, min(G_k)\varepsilon(G_k) \tag{3.3}$$

3.2.5 MiniMax method

The minimum value of the maximum of K^{th} consensus level ($j\varepsilon\{1, ..., N\}$) and output of each of the methods are used to select the optimum position for ground truth image. Which is expressed as follows,

$$G_k = max\{V_{j,k}|1 \leq j \leq N\} \tag{3.4}$$

$$k_{optimum}, min(G_k)\varepsilon(G_k) \tag{3.5}$$

The output of each method is shown in figure 3.3. The choice of best reference image using these two (minimean and minimax) methods is an essential step for performance evaluation.

3.3 Performance measures

The edge detection algorithms are based on different criteria thus, they are sensitive to distinct edges. To judge the efficacy of the edge detection methods Canny proposed three criteria as discussed in chapter 2. Therefore, performance measures are required to find an edge detector which satisfies these criteria. Four different performance indices have been considered here to server this purpose.

- Edge Miss match Error (EMM)
- figure Of Merit(FOM)
- F-measures
- Precision-Recall curve (PR curve)

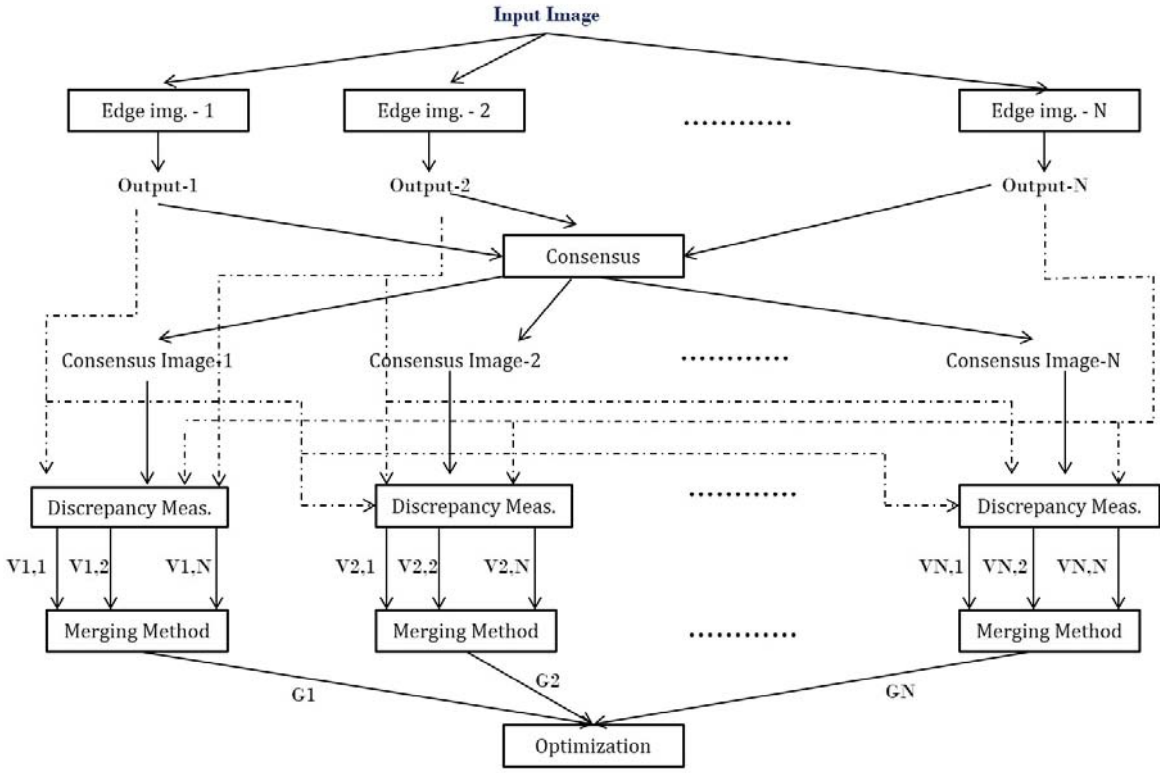
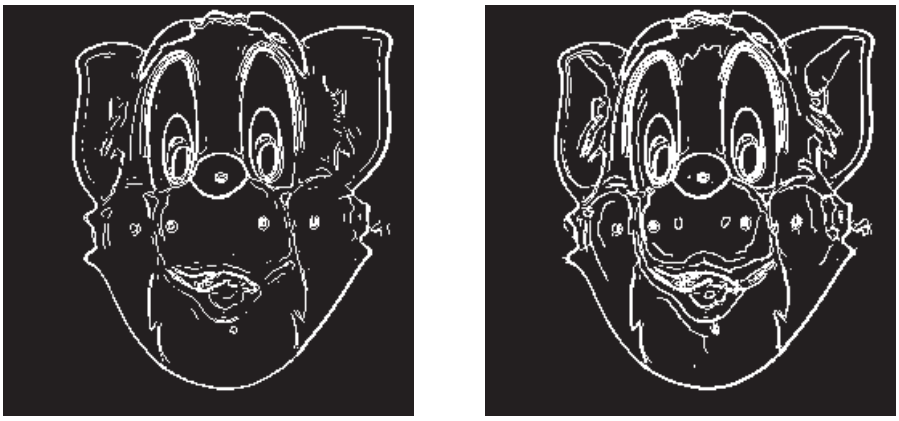


Figure 3.1: Frame work for generation of consensus ground truth

(a)

Figure 3.2: Input image



(a) (b)

Figure 3.3: Consensus ground truth by using(a) minimean, (b) minimax method

All the selected metrics require reference image for evaluation, which is generated using discussed consensus approach (see Section 3.2.3 for more details).

3.3.1 Edge Missmatch Error (EMM)

Edge Missmatch Error (EMM) finds the measure of change in the location of edge pixels in the detected edge image with respect to the reference edge truth image.

$$EMM = 1 - \frac{CE}{CE + W[\sum_{k\varepsilon(EO)} \delta(K) + \alpha[\sum_{l\varepsilon(ET)} F(l)]]} \tag{3.6}$$

Here,

⋆ CE (Common Edges) - The number of common edge pixels between reference ground truth image and edge image.

⋆ EO (Excess Original Edge) - The number of excess edge pixels in reference ground truth image which are not present in edge image.

⋆ ET (Excess Thresholded Edge) - The number of excess edge pixels in edge image which are not present in reference ground truth image.

⋆ $\delta(K)$- Distance function

$$\delta(K) = \begin{cases} |d_K|; & f|d_k| \leq maxdist \\ D_{max}; & otherwise \end{cases} \tag{3.7}$$

⋆ d_k- The Euclidean distance between the k^{th} edge pixel to the complementary edge pixel within maxdist.

⋆ maxdist = 0.025 N

⋆ $N = \sqrt{N_{row} * N_{coloumn}}$

⋆ $D_{max} = 0.1N$

⋆ $w = \frac{10}{N}$

⋆ $\alpha = scalingfactor(2)$

3.3.2 figure Of Merit(FOM)

Pratts figure of Merit (FOM) is appropriate measure to identify the false positive error. This measure lacks in are having the disadvantage as there is no proper theoretical justification [19] and it is also not sensitive to a false negative type error. However this index provides discrepancy measure for edge localization, estimating minimum distance from expected edge point. It is normalized in between 0 and 1, which is formulated as follows:

$$F = \frac{1}{max\{I_{ref}, I_{edge}\}} \sum_{i=1}^{I_{edge}} \frac{1}{1 + \alpha d^2(i)} \tag{3.8}$$

Here

⋆ I_{ref} -The no of edge points in reference ground truth image.

⋆ I_{edge} -The no of edge points in detected edge image.

⋆ α -Scaling constant=$\frac{1}{9}$

⋆ d - Separation distance between the edge point in the reference ground truth image and detected edge image.

Eq..eq:ch3:Performance evaluation indicates the separation distance 'd' is inversely proportional to FOM. For smeared edge the displacement of pixels from its original position increase which causes an increase in separation distance and the corresponding value of FOM decrease. Thus, the edge detectors with a higher value of FOM are well localized.

3.3.3 F-measure

F-measure or F-score or F_1metric is the weighted harmonic mean of precision and recall. It is used to indicate the classification accuracy.

$$F - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.9}$$

Here ,Precision or sensitivity or True Positive Rate is the ratio of the no of pixel which are defined as edge pixel in both reference and detected image to the total no of edge pixel in detected image.

$$Precision = \frac{TP}{TP + FP} \tag{3.10}$$

Recall - on the other hand it is the ratio between the number of pixel which are defined as edge pixel in both reference and detected image to the total number of edge pixel in reference image.

$$Recall = \frac{TP}{TP + FN} \tag{3.11}$$

The confusion matrix for the F-Measure as shown in figure 3.4, yields detector performance.

3.3.4 Precision Recall (PR)Curve

Precision Recall (PR) curve represents the relationship between the recall and sensitivity for a large dataset/database. Binary decision based dataset

		Detected	
		Positive	Negative
Actual	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

Figure 3.4: Confusion matrix

may be well classified using both Receiver Operating Curve (ROC) and/or Precision Recall (PR). However literature suggests that PR curve more effectively classify the skewed dataset, i.e. data with negative counts much more than positive counts [14]. Thus a large change in false positive can raise a very small change in False Positive Rate (FPR). Unlike ROC curve, Precision Recall curve compares false positive with true positive instead of true negative. This includes a large number of negative examples in the algorithm's performance. Therefore, PR curve would provide better evaluation for different edge detector than ROC curve.

3.4 Discussion

This chapter includes different type of approaches for the ground truth generation. It also covers application and limitations of these approaches. The ground truth edge map, thus obtained from consensus method is used for performance evaluation of different edge detectors using discussed quantitative evaluation metrics.

Chapter 4

Real-time Canny Edge Detection System Design

4.1 Introduction

This chapter describes the design of a real time image processing for Canny edge detector system which make use of the test bench developed in chapter 5. The design and simulation was done for a single frame. The targeted system should acquire frames from an acquisition system and should detect edges in them.

4.2 Architecture Selection/Parallel Algorithm Development

4.2.1 Architecture Selection on FPGA

In the FPGA based circuit design, estimation of different system performance parameters is required. The first step is the selection of the FPGA system architecture. A Canny based edge detection algorithm is based on sequential operation but it can also be designed by considerable amount of parallelism in it. Each module of canny can be implemented in parallel. Thus to effectively utilize this parallelism, an architecture which is completely parallel in nature will be selected. A standalone FPGA architecture is the best choice for this

design, as it is completely parallel.

4.2.2 FPGA Computational Architecture Selection

The next step is the selection of FPGA memory (computational) architecture. This design involves buffers at input and output end, for storing the incoming input image pixel values and processed output values. Thus the incoming pixels can not be processed on the fly. The best method of accessing data from a buffer is random access processing. Path between the input and output buffer are continuous at the same time. Therefore, there is no need of storing intermittently. Hence, stream processing can be used in this section. Thus the whole design is a combination of random access processing and stream processing.

4.2.3 Selection of FPGA Mapping Techniques

The FPGA mapping techniques is used for achieving desired system performance. In this design, pipe-lining and caching has been used for accelerating the overall system performance without loss of pixels. A pipeline is implemented to accelerate the processing of the pixels. The Gaussian mask, SOBEL mask, non maxima suppression is implemented in pipe-line. Each pipe line includes three stages. The result will be stored to the output cache memory in the third stage. While the current pixel is being stored to output cache, the result of the next pixel will be calculated simultaneously in the second stage of the pipe. At the same time, a third pixel will be fetched in first stage of the pipe. This pipelining of data helps the system to process many pixel at the same time.

Selection of Cache Memory

Caching is used to improve system latency. In this process a four pixel wide cache is applied to avoid frequent reading of data from the main memory. To

find out the result at the central pixel, nine neighboring pixels (3 rows of 3 pixels) are needed . But as normally the size of cache memory is defined in powers of two so a 4 pixel wide cache memory is setup to store one row. Three such memories are designed to store three neighboring rows i.e. previous row, current row and the next row. In particular caching is implemented by row buffering technique.

4.3 Over all architecture

The over all architecture of the Canny edge detection system is as described in chapter 2. The architecture is divided into four parts.

- Smoothing
- Gradient calculation
- Non Maxima Suppression
- Hysteresis Thresholding

4.3.1 Smoothing

The entire circuit of smoothing operation can be divided into four parts [20].

- Buffering
- Indexing
- Caching
- Pipeline

The data from the MATLAB output hex file will be buffered in the internal frame buffer, with one pixel in every clock pulse. This data will be then routed to cache memory through appropriate control signals. To calculate a smoothed pixel, eight neighboring pixels has to be fetched from buffer. Four

pixels will be fetched at a time from the buffer and will be stored in a four byte cache. Four pixels of previous row, present row and next row will be buffered in to three different cache memories, with each memory has a size of four bytes. At the end of first row of buffer, the test bench will generate a horizontal synchronous pulse, it will disable the data valid signal to represent the data is invalid. After the completion of three rows this will enable the data path between buffer and the cache memory corresponding to present row (present row cache). In a similar way data will flow from buffer to the cache memories corresponding to current and next rows. The addressing for the different row is as shown in figure 4.4. When the three rows are filled, by using shifting operation pipeline acquire data. The gaussian mask is applied in pipelining as in figure 4.5a-4.6b. Path from pipeline to the output cache memory will not be enabled unless the input row cache memories are full. The result will be stored in the output gaussian buffer.

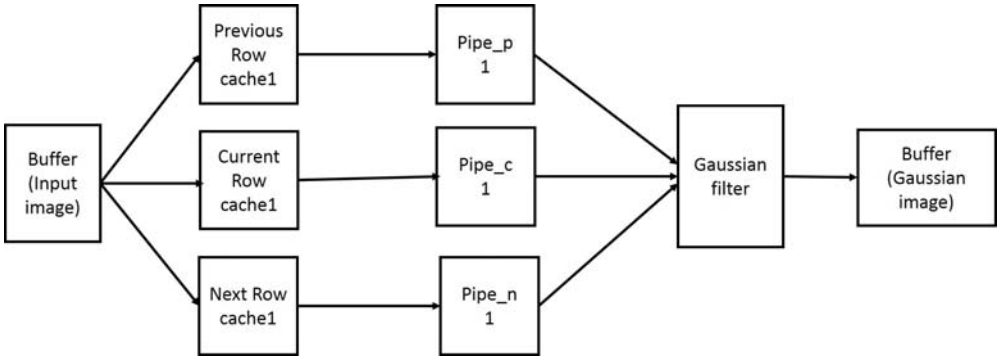
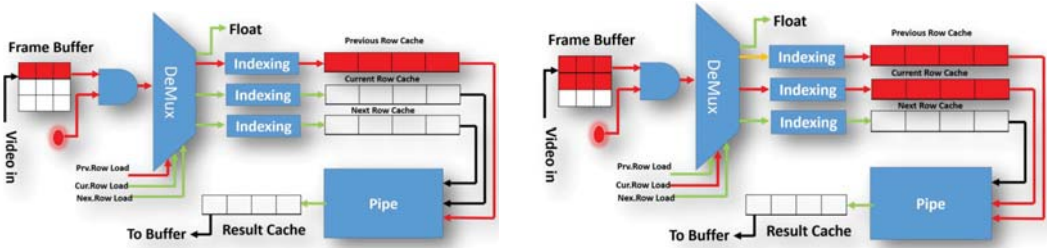


Figure 4.1: Architecture for Gaussian smoothing

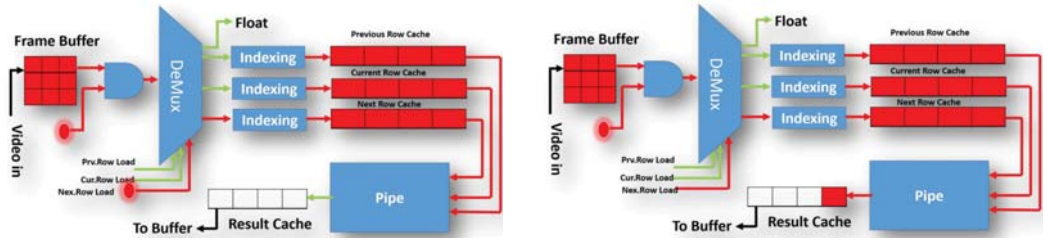
4.3.2 Gradient and magnitude calculation

As explained in Chapter 2 the next step of Canny is gradient and magnitude calculation. For this Sobel mask in x and y direction is implemented in pipe line [21] in similar way as explained in above section for Gaussian mask. The architecture for these calculation is as show in figure 2.3. Data from Gaussian buffer is divided into three cache memories, and shifted towards pipe with



(a) (b)

Figure 4.2: (a) Previous Row buffering, (b) Current Row buffering



(a) (b)

Figure 4.3: (a) Next Row buffering, (b) Result Row buffering

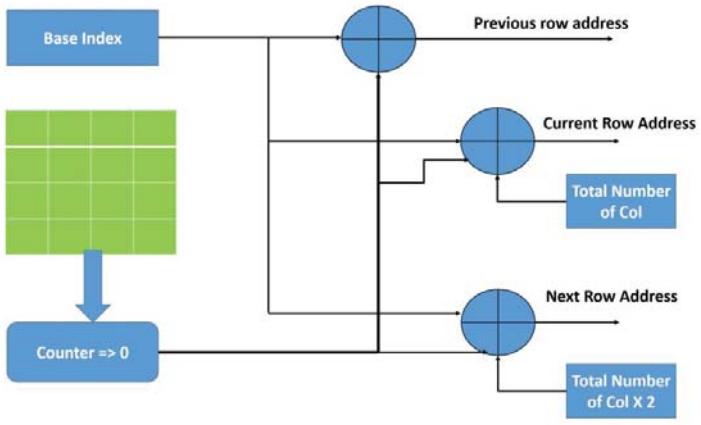
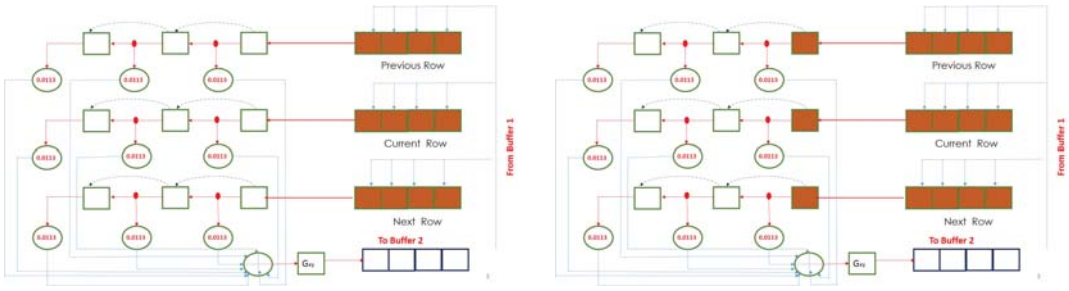


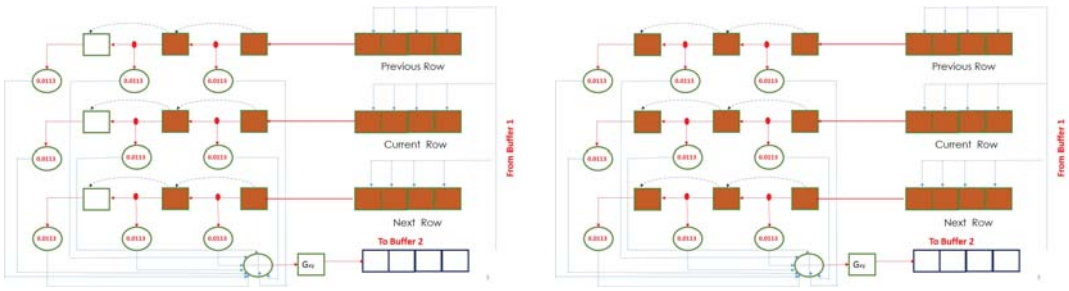
Figure 4.4: Indexing Circuit

increase in each clock. After the Sobel operation in both the direction in pipeline the result gradient in x and y direction is used for gradient magnitude and direction calculation. The steps for magnitude and angle calculation is included with the architecture of non maxima suppression as explained in next section.



(a) (b)

Figure 4.5: Data flow of pipeline in first three clock



(a) (b)

Figure 4.6: Data flow of pipeline in next two clock

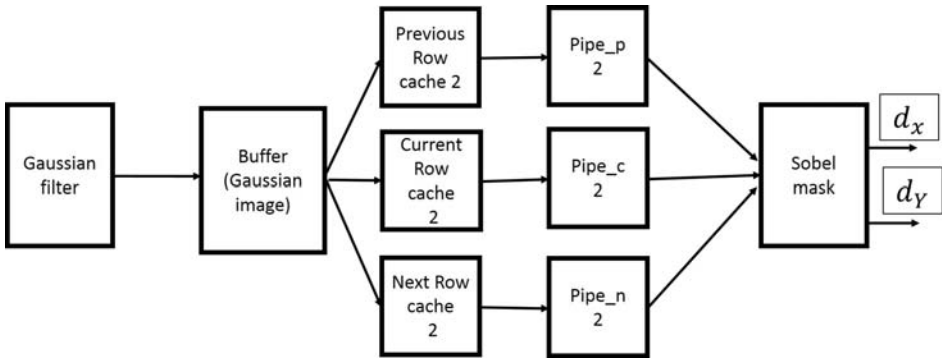


Figure 4.7: Architecture of d_x and d_y calculation

4.3.3 Non-maxima calculation

The architecture for non maxima suppression is in figure **??**. The gradient magnitude and direction is calculated by using d_x and d_y value in previous section. For the angle calculation look up table is used. The angle is approximated to the four predefined gradient direction. Then the comparison

of the current pixel with its two neighborhood along the direction is done. If the magnitude of the current pixel is less than any of the magnitude of its neighbor along the direction replace the current pixel with zero i.e. no edge. Else the current pixel is marked as edge point.

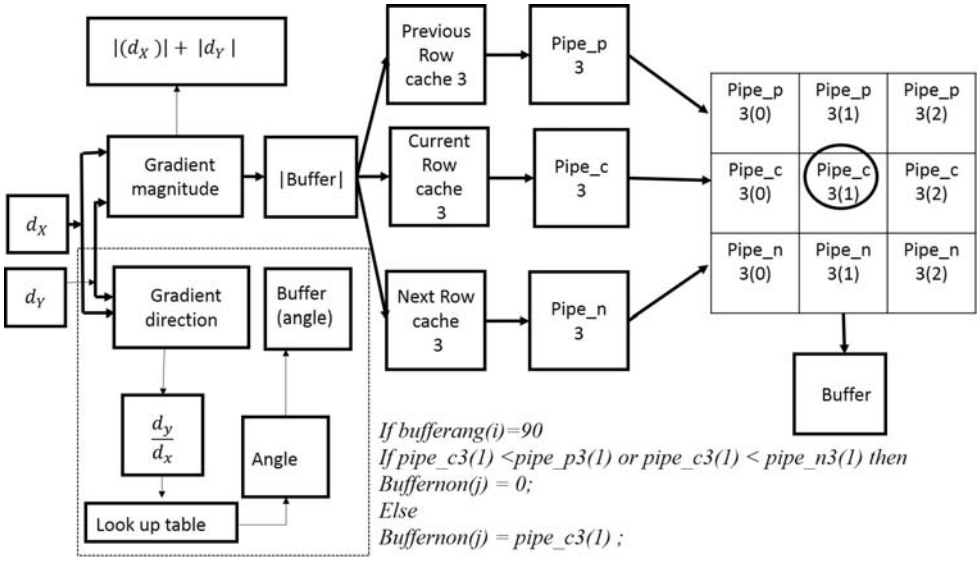


Figure 4.8: Architecture of Non-maxima calculation

4.3.4 Hysteresis calculation

In hysteresis thresholding we choose two threshold values as discussed in chapter 2. In the output buffer of previous step we apply these two thresholds by using comparator as shown in figure fig:ch5:Hysteresiscalculation1. These threshold values segregate the buffer into two image $f_1(x, y)$ and $f_2(x, y)$ pixel is results in two image. With the use of consecutive OR and AND gate as, false edge points are removed by using following steps

- If the central pixel in $f_1(x, y)$ is 1 then it is directly taken as a true edge pixel.
- If the central pixel in $f_1(x, y)$ is 0, and if any of the neighbor in $f_1(x, y)$ of the current pixel is 1 and central pixel in $f_2(x, y)$ is 1, then it is taken

as a true edge pixel, else discard the pixel.

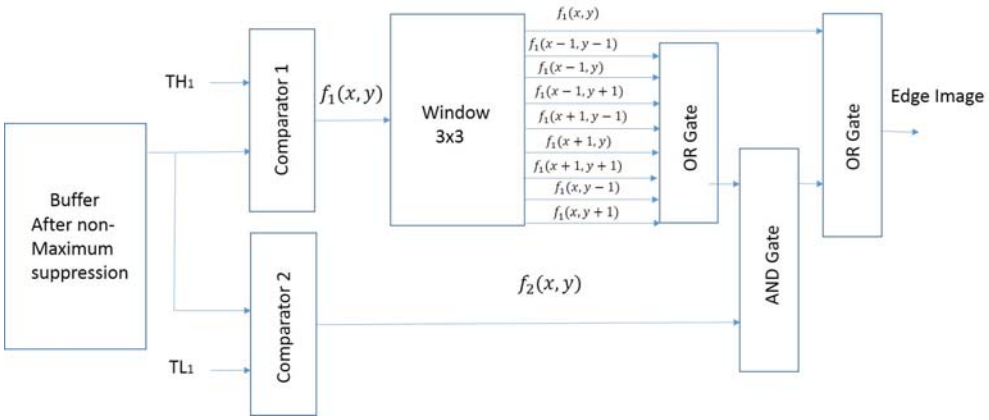


Figure 4.9: Architecture of Hysteresis calculation

4.4 Discussion

This chapter include the architecture design of Canny edge detection algorithm for including considerable amount of parallelism in it . For the design of each section buffering, caching and pipeline processing is used. As in Canny every step is dependent on the previous step result. In this design each module is working in parallel after some initial delay. These parallelism in design accelerate the overall architecture speed.

Chapter 5

Design of a VHDL Test-bench for Canny edge detection

5.1 Introduction

A real-time system process information and generate a response within a specified time limit, else risk severe consequences, including failure. In image processing the real-time system must process an input image to produce certain output image such that it preserves desired features for down stream applications within a certain time limit. To gain maximum performance, these image processing systems are designed using parallel processor and the architectures are implemented in various hardware platforms.

5.2 Need for the Test-Bench

For hardware implementation of any design problem, initially it must be validated and verified in software platform. Once hardware is designed it will be difficult and time consuming task to fix any problem. Hence in software we need to develop an architecture for simulation of real-time systems. In this thesis, we have developed a test bench and simulated it in VHDL for edge detection of an image using Canny's method. The reported architecture

may serve as the prototype for its hardware implementation in FPGA.

5.3 Model for an Image Processing System

framework for the representation of real time image processing system is given in figure **??**. Initially image is acquired from the sensors of camera unit.The preprocessing is achieved by IP core using sequential processors like MC and DSP. Preprocessing steps may also includes image filtering, smoothing of image for noise reduction and color to gray conversion. The image frames thus stored will be accessed by a prime image processor, based on applications, which may be an FPGA-based parallel processor, for further processing. In this processor any image processing algorithm can be implemented. After the completion of assigned task output will be stored in frame buffer. This buffer and display are connected to same IP core, from which data can be sent to any output device.

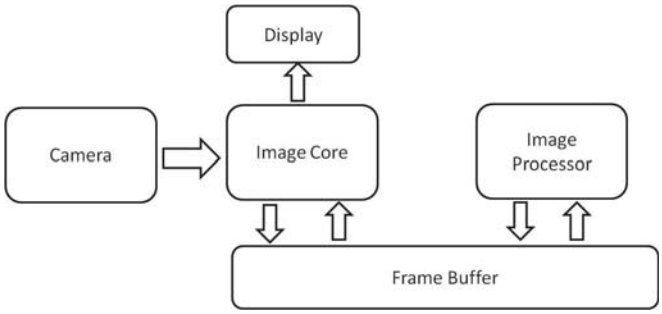


Figure 5.1: Model of Image Processing System

5.4 Image Representation

As the Fig.(5.2) indicates the composite image signal contains image intensity values as well as certain synchronization signals. A horizontal and vertical synchronous pulses are used to indicate end of row and end of the image, respectively. Moreover any incoming data is validated using a data valid signal. This can be checked using a high pulse.

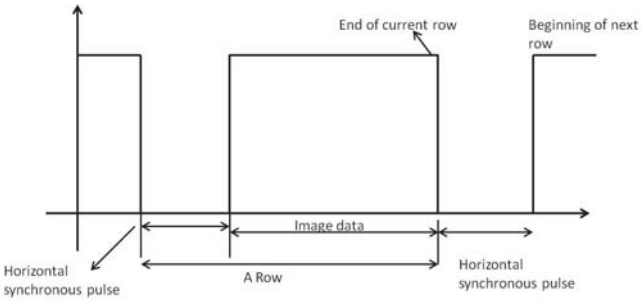


Figure 5.2: Image Representation in analog form

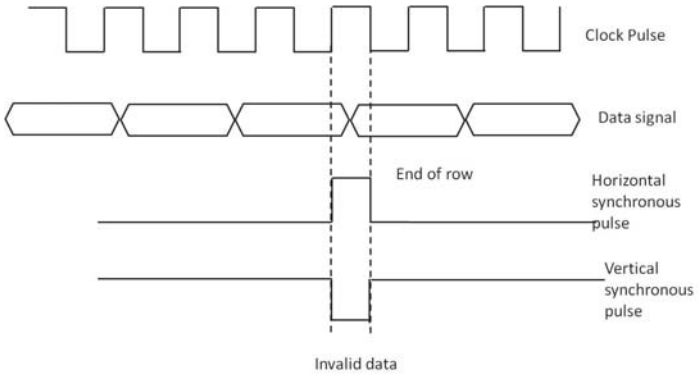


Figure 5.3: Image Representation in digital form

5.5 Image Processing Test Bench

The image processing test bench must be able to read an input image file, generate clock pulses for various operations, generate synchronization pulses and make the image accessible to the main processor where necessary operation is performed on the image. Since images cannot be accessed directly in VHDL, we need to generate a text file from external program (MATLAB is used here) which can represent the image in readable form to VHDL. The same external MATLAB program is used for the generation of output image from the output hex file, generated in VHDL. The overall process is shown in block diagram with three important modules of the test bench is as shown in figure 5.4

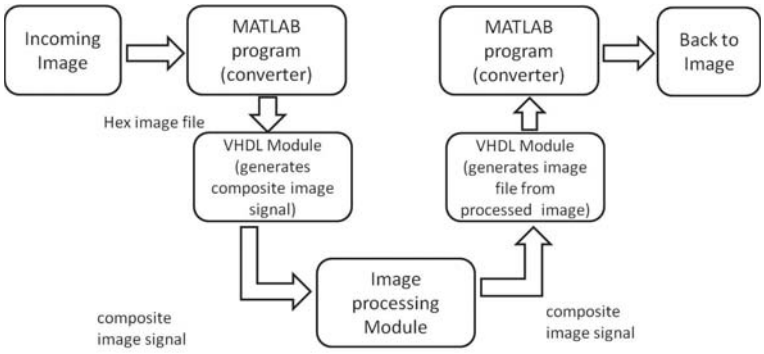


Figure 5.4: Block diagram of a image processing test bench

5.5.1 Image to Hex-Image Converter

Since VHDL cannot access the images directly, this MATLAB program will convert the incoming image frame in to hex format image file where each intensity value will be written in hexadecimal format.

Hex-Image File

This image can be accessed by writing the intensity values of the image into a text file. The text file can be generated by an external program in MATLAB. Each pixel value or the intensity value of the image will be converted and written in to hexadecimal format. It will be stored as a sequence of 2 continuous characters.

For example the intensity value 8 will be converted into hex value stored as 08. Similarly the intensity value 255 will be stored in hex format as FF. All the intensity values from 0 to 255 is converted to their respective Hex-values, and coded between 00 and FF. A ‘, character is written at the end of each row. A ‘* character will be placed at the end of each image frame. This conversion and coding scheme and finally writing down to a text file is shown in Fig.(5.5). The image file with hex values of pixels is generated for a 256 x 256 Lena image.

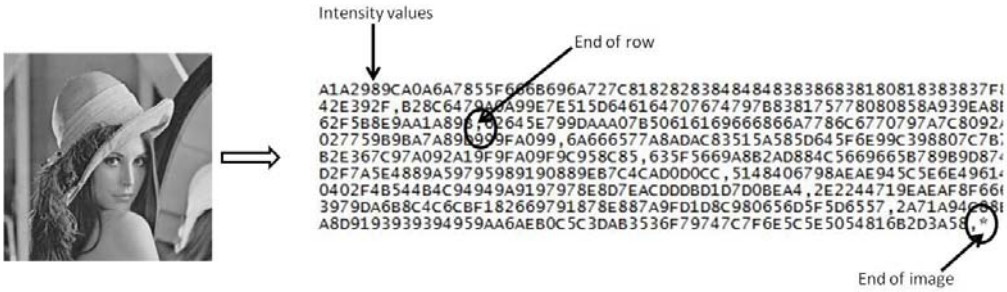


Figure 5.5: Coding of image into a hex image file

5.5.2 VHDL Camera Module

This module is a program written in VHDL. These programs were written and developed in ModelSim 10.4a platform. This module takes the image pixel values from the hex file generated. The hexadecimal values will be converted into standard logic vector of 8 bit length. In addition, it generates clock pulses for operation of the FPGA processor and display driver. It generates clock pulse of 100ns. After every 100ns the clock pulse will change its state. Whenever the rising edge of clock occurs, the program reads a character from input hex file. It checks the character then. If it is a ‘,’, then a horizontal synchronous pulse is generated and if the character is ‘*’, then a vertical synchronous pulse is generated. Each of the horizontal synchronous pulse vertical synchronous pulse are of one clock duration. If the encountered character is a hexadecimal number then a ‘data valid’ signal is generated and it will be converted to standard logic format. Single bit of hexadecimal number will be converted to 4 bit length standard logic vector. Thus for 2 bits of hexadecimal number, this module generates 8 bit length standard logic signal representing the intensity value of the image pixel. The flow chart of the camera module is shown in Fig.(5.6).

5.5.3 Output Display Module

The Display of output image can be attained by writing the output image values into a text file. This text file can be read by an external MATLAB

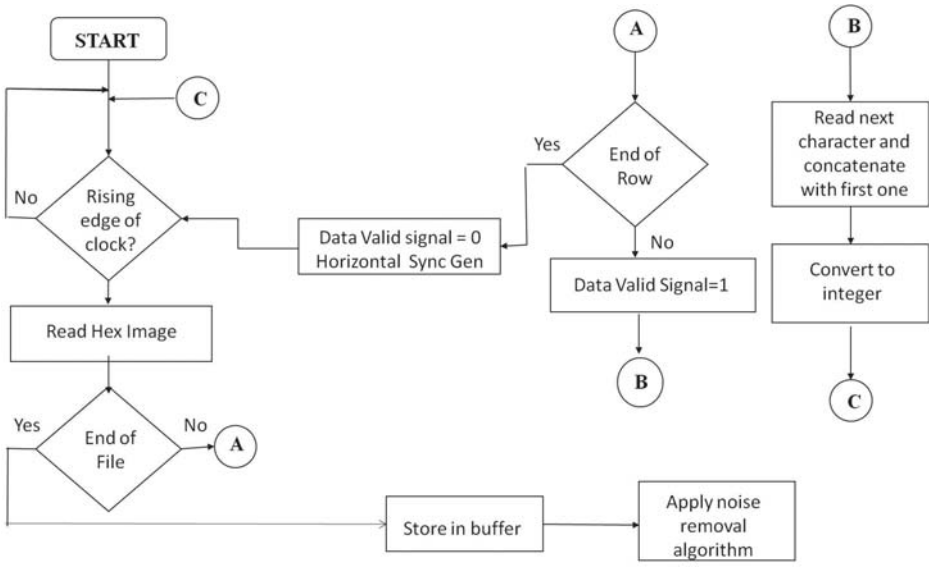


Figure 5.6: Algorithm for generating composite image signal from hex file

program. It generates the output image from the text file and thus displays the output.

5.5.4 Output Signals from Test Bench

The main output signals present in camera module of VHDL are as follows:

- A clock pulse
- Horizontal synchronization pulse
- Vertical synchronization pulse
- Data valid signal

The simulation was performed in ModelSim 10.4a and the generated results are shown in Fig.(5.7).

Clock Pulse

The clock pulse is designed to change its state after every 100ns. Depending on the application the clock time period can be modified. By default clock

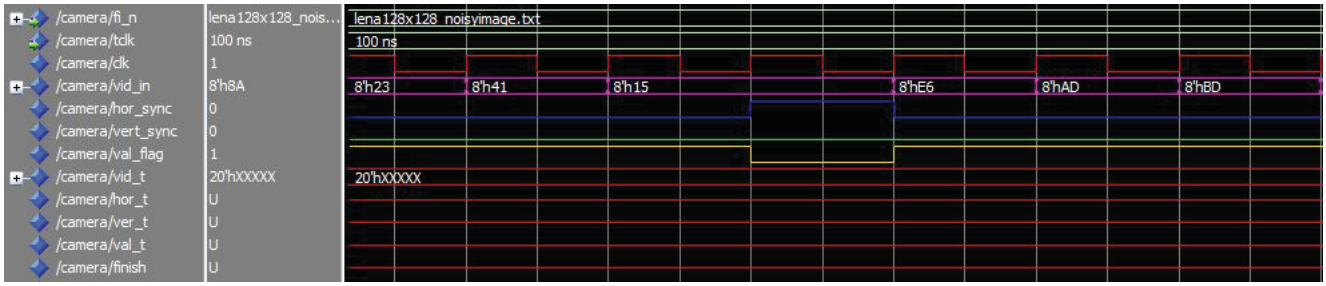


Figure 5.7: Simulation results

pulse is designed to toggle its state in every 100ns. As the application demands, this value can be modified to meet the specifications. All the signals in camera module, Edge main module and display module and other events will be synchronized with the clock pulse. The Fig.(5.7) shows the generated clock pulse in red color.

Horizontal Synchronous Pulse

Whenever there is an end of the row a horizontal synchronous pulse will be generated. It is of one clock pulse duration. Total number of horizontal synchronous pulse generated will be equal to the number of rows in our input image. The signal is shown in blue color in the waveform in Fig.(5.7).

Vertical Synchronous Pulse

Whenever VHDL detects that there is an end of the frame a vertical synchronous pulse will be generated. It is also of one clock pulse duration. Total number of vertical synchronous pulse generated will be equal to the total number of frames. The signal is shown in green color in the waveform in Fig.(5.7).

Data Signal

This is an image data signal shown in pink color. The data signal is present in standard logic vector format, and it will be sent to the image processor. The

data signal is of length equals to 8 bit Fig.(5.7). The camera module changes the data from hexadecimal to standard logic format and display module will get same data signal. After that it will be converted in to hexadecimal format for writing into output file.

Data Valid Signal

The status of the data signal is informed by the data valid signal. It goes low if an invalid data is detected otherwise for a valid data signal it remains high. Whenever a horizontal and vertical synchronous pulse are high,the data valid signal signal goes low. That means on the occurrence of a ',' or a '*' data valid signal is low. Signal is shown in yellow color in the Fig.(5.7).

5.6 Discussion

Before any hardware implementation it is essential to validate and verify any real time model in simulation platform. In this chapter we discuss about the implementation procedure for image analysis in VHDL platform. The developed test bench can be simulated for real-time image acquisition system.

Chapter 6

Results and Discussion

In this thesis, we have compared various edge detection algorithm. Here, three different categories of edge detection techniques are considered that includes Sobel, Roberts, Canny, Prewitt, Phase congruency and LOG operator (as discussed in chapter 2). For the comparison, different performance indices such as: edge miss match error (EMM), figure of merit (FOM), F-measure and precision recall curve (PR) curve is used. These performance measures require reference image for evaluation. Consensus of different edge detectors is used for automatic generation of ground truth image which takes two merging method, minimean and minimax.

6.1 Results

6.1.1 Experimental setup

In this context four different databases with more than 1500 images of different human faces, natural scenes, buildings and objects containing different textural characteristic, with various percentage of homogeneous regions, edges, and details are considered for the experiment. More detail about each database are given in Table.6.1. For unification purpose, all the images in these databases are cropped to size 256×256 pixels.

Table 6.1: Database used for experimental validation

Database	Original Image Type	No. of Images	Image Resolution	Other Specification
Airplane and Leaves image	Airplane and Leaves image	1260	602×402	1074 Airplane and 186 Leaves image
TID 2008 Database	Natural and Artificial Images	25	512×384	24 Natural images and 1 Artificial image
Zurich Object Database	Object Images	381	320×240	115 Building and 269 Object Images

6.1.2 Consensus based reference image generation

As discussed in chapter 3 for the comparison in various edge detectors there is a requirement of reference image. The reference edge map as shown in figure 6.1 is generated considering minimean and minimax method, respectively. MATLAB based Graphical user interface(GUI) as shown in figure was developed as a part of this work for quantitative evaluation of edge detection. This GUI has two options, we can either choose one image or multiple images at a time. For single image it generate the consensus reference image, edge map of all detectors and three performance measure for comparative evaluation. We also recorded performance measures for different image databases using multiple image option in GUI. These measures are used for quantitative comparison.

6.1.3 Performance measure on different data base

Initially all the considered metrics are normalized between [0,1], where 0 indicates the reference edge map and the test edge map are same while 1 represents otherwise. The overall performance measure for each image database is calculated using Average Performance index (API) for comparative analysis. Which is defined as follows:

$$API = \frac{EMM + F - measure + FOM}{3} \tag{6.1}$$

(a) (b)

Figure 6.1: Consensus based reference image using, (a) minimean and, (b) minimax merging method

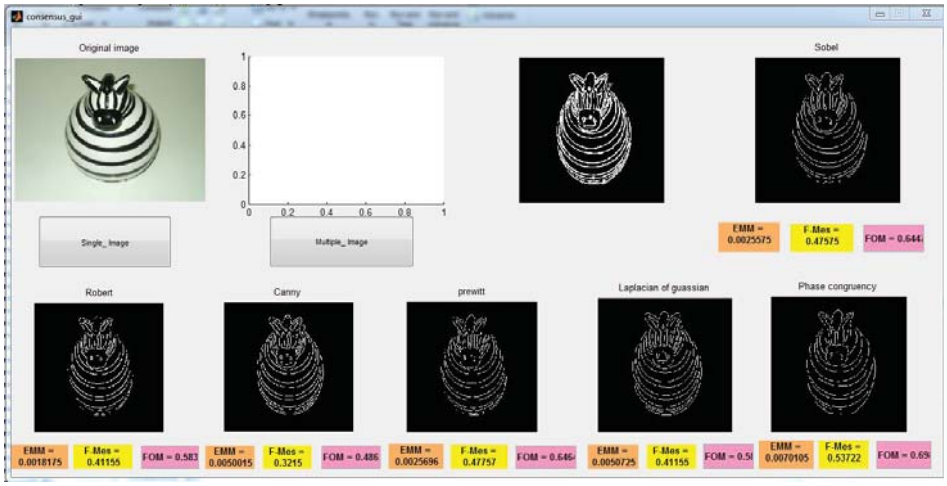


Figure 6.2: MATLAB based GUI for edge detection evaluation.

Performance measure results for each data base is shown in Table. 6.2-6.9.

Table 6.2: Performance measure for Caltech airplane database with minimean merging

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.4828	0.4815	0.4813	0.5766	0.4024	0.7593
EMM	0.0481	0.0476	0.0145	0.0523	0.0031	0.0160
FOM	0.5938	0.5926	0.3075	0.6271	0.3483	0.7826
API	0.3749	0.3739	0.2677	0.4186	**0.2512**	0.5193

Table 6.3: Performance measure for Caltech leaves datasbase with minimean merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.5378	0.5389	0.4485	0.5884	0.4476	0.6636
EMM	0.0714	0.0902	0.0729	0.0581	0.0077	0.1185
FOM	0.6468	0.6493	0.2681	0.6436	0.3888	0.7096
API	0.4186	0.4261	**0.2631**	0.4300	0.2813	0.4972

Table 6.4: Performance measure for TID2008 with minimean merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.5194	0.5191	0.5202	0.6599	0.3640	0.7457
EMM	0.0277	0.0139	0.0137	0.0563	0.0129	0.0322
FOM	0.6442	0.6471	0.1749	0.7191	0.2365	0.7743
API	0.3971	0.3933	**0.1749**	0.719	0.2365	0.7743

Table 6.5: Performance measure for Zurich Object Database with minimean merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.4250	0.4238	0.4731	0.5212	0.4330	0.6370
EMM	0.0096	0.0071	0.0129	0.0322	0.0131	0.0037
FOM	0.4988	0.4994	0.2767	0.5147	0.3261	0.6268
API	0.3111	0.3101	**0.2542**	0.3560	0.2574	0.4225

Table 6.6: Performance measure for Caltech airplane datasbase with minimax merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.5014	0.4998	0.4442	0.5774	0.4835	0.7676
EMM	0.0006	0.00066	0.0020	0.0009	0.00103	0.0008
FOM	0.5833	0.5821	0.3891	0.6201	0.4312	0.7886
API	0.3617	0.3608	**0.2784**	0.3994	0.3052	0.5190

Table 6.7: Performance measure for Zurich Object Database with minimax merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.5904	0.5910	0.3449	0.6004	0.4805	0.7107
EMM	0.0012	0.0011	0.0040	0.0010	0.0010	0.00115
FOM	0.7386	0.7392	0.4924	0.7444	0.5868	0.809
API	0.4433	0.4437	**0.2804**	0.4496	0.3561	0.5069

Table 6.8: Performance measure for Caltech leaves database with minimax merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.5422	0.5430	0.4315	0.5816	0.5125	0.6647
EMM	0.0110	0.0110	0.0042	0.0085	0.0077	0.0056
FOM	0.6316	0.6341	0.3551	0.6246	0.4541	0.6928
API	0.3949	0.3960	**0.2635**	0.4049	0.3247	0.4543

Table 6.9: Performance measure for TID2008 database with minimax merging for reference image generation

	Sobel	Prewitt	Canny	Roberts	LOG	Phase Congruency
F-measure	0.4294	0.4267	0.6151	0.5933	0.5619	0.6689
EMM	0.00061013	0.000666	0.0018137	0.00079491	0.00223	0.0077
FOM	0.3493	0.3476	0.2519	0.4301	0.2561	0.5553
API	0.2597	**0.2583**	0.2896	0.3413	0.2734	0.4106

Table 6.10: AUC calculation for Precision Recall curve using minimean method

	Caltech airplane datasbase	Caltech Leaves dataset	TID2008	Zurich Object Database
Sobel	0.7178	0.7198	0.5898	0.6787
Prewitt	0.7227	0.7072	0.5506	0.6817
Robert	0.5694	0.6149	0.5190	0.6061
LOG	0.5982	0.6492	0.5031	0.6134
Phase Congruency	0.4532	0.5213	0.4482	0.5189
Canny	**0.7368**	**0.7371**	0.6214	**0.6832**

6.1.4 Precision Recall curve

Precision recall (PR) curve is a graphical approach, which represent sensitivity verses precision. PR curve for minimean and minimax method is as shown in figure 6.3 and 6.4. Area under the curve is also calculated which is shown in Table 6.11 and 6.11. AUC lies between 0 and 1 where, 0 is considered as worst and 1 is as best value. If the PR curve of any edge detector is greater than 0.5 then it is considered as a better edge detector. More the AUC more the performance of considered detector.

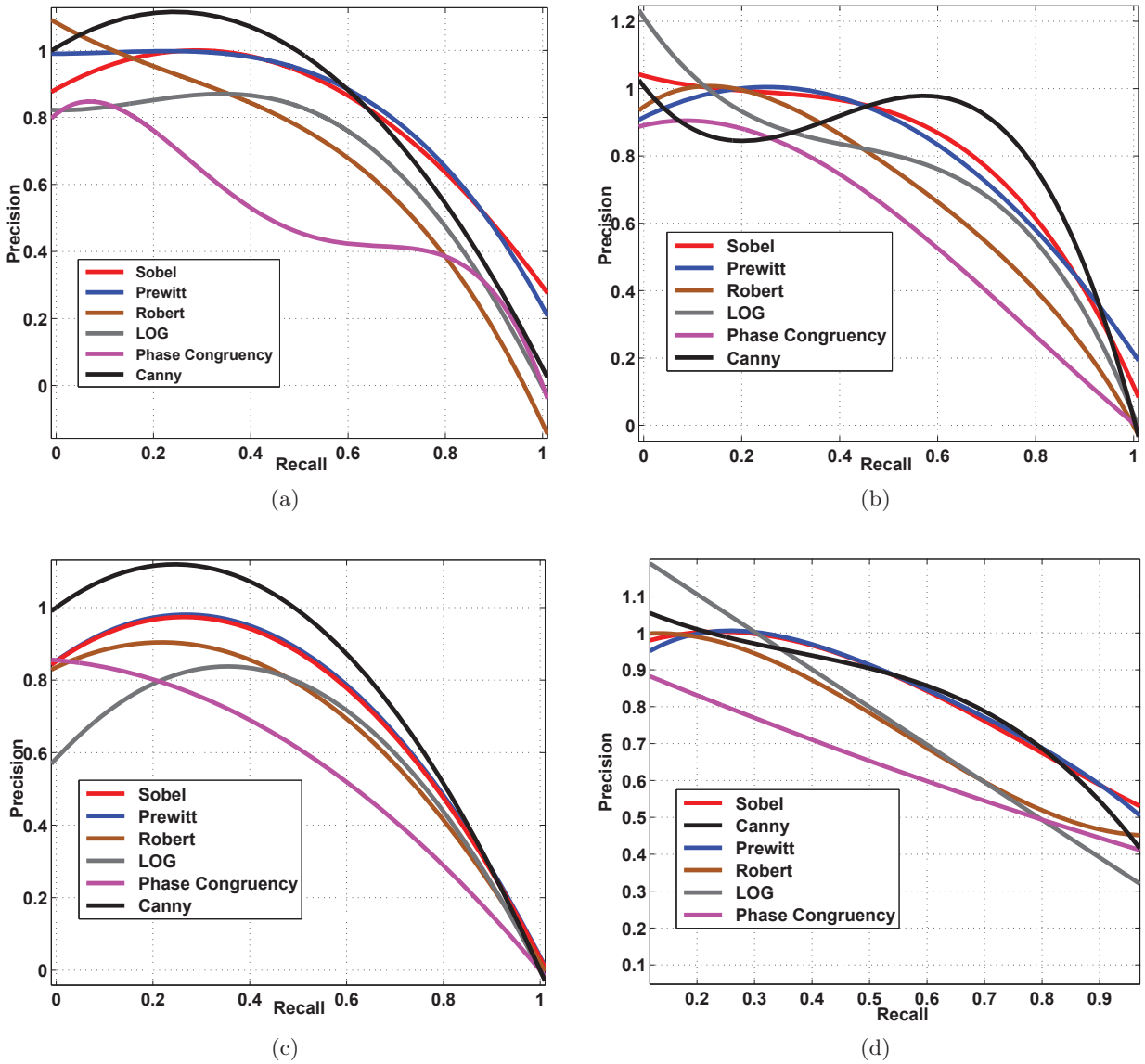


Figure 6.3: Comparison of different edge detector for minimean method using PR curve on (a)Caltech airplane database[——ref——], (b)Caltech leaves database,(c)TID2008, (d)Zurich Object Database

6.1.5 Edge detection in VHDL

From the results of Performance measure and PR curve for minimean and minimax on each database, we found that Canny edge detector is a consistent operator for edge detection. Thus for implementation of canny edge detector an architecture is developed in VHDL for hardware realization in FPGA. The

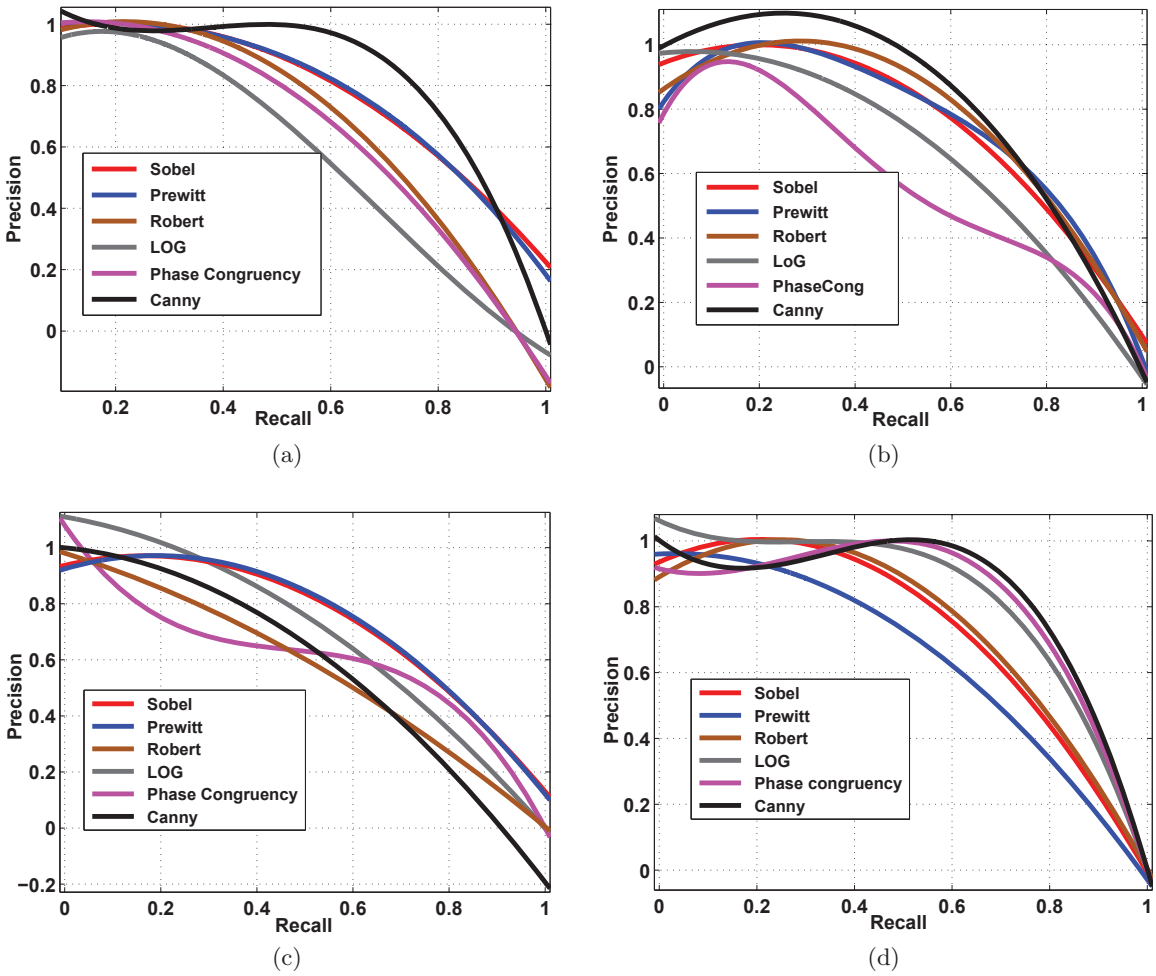


Figure 6.4: Comparison of different edge detector for minimax method using PR curve on (a)Caltech airplane datasbase, (b)Caltech leaves datasbase,(c)TID2008, (d)Zurich Object Database

developed architecture adopts Sobel's mask for gradient magnitude and angle calculation for Canny edge detector. For the simulation, as in VHDL we can not read image directly thus we have used ModeSim PE Student edition 10.4a along with MATLAB 2012 is used to read an image in VHDL for simulation of Sobel's architecture [**?**]. MATLAB converts the image file format to hex file, that serves as input in VHDL. The processed image (in hex format) from VHDL is converted back to binary image using MATLAB. Initial attempt for edge detection using Sobel's operator is simulated using VDHL. The result as shown in figure 6.6, indicates the potential of parallel processing in image

Table 6.11: AUC calculation for Precision Recall curve using minimax method

	Caltech airplane datasbase	Caltech Leaves dataset	TID2008	Zurich Object Database
Sobel	0.6828	0.6595	0.5842	0.5912
Prewitt	0.6824	0.6703	0.5741	0.5298
Robert	0.5972	0.6848	0.5143	0.6215
LOG	0.5123	0.5660	0.5283	0.6878
Phase Congruency	0.5731	0.5210	0.4548	0.6961
Canny	**0.7532**	**0.7193**	0.5083	**0.7105**

analysis.

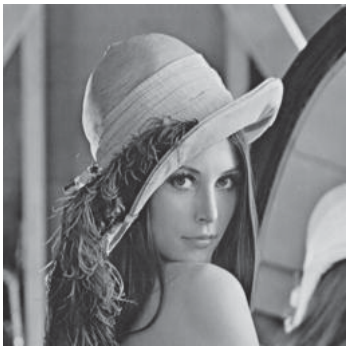


Figure 6.5: Input image

(a) (b)

Figure 6.6: Ege detction by Sobel method, (a) MATLAB output, (b)VHDL output image

6.2 Discussion

Experimental setup is created as shown in Table 6.1. The results indicate that the performance of edge detection algorithm varies with the image i.e.

behavior of detector depends totally on selection of images. Results also illustrate that point edges are more prominent and easily distinguishable. The images with less textural information provides better edge map. Unlike, we have considered both minimean and minimax methods for edge evaluation. Since Canny uses smoothing for noise removal, nonmaxima suppression for edge localization and hysteresis for false edge extraction, it is more robust for different type of images. The architecture for Canny edge detector uses Sobel mask for gradient magnitude calculation in VHDL. The result for Sobel operator illustrate that the time efficient processing of FPGA based parallel processor.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this thesis, we have studied different types of edge detection techniques and evaluated these methods using four performance measures. The need of reference edge map for quantitative evaluation using EMM, FOM, F-measure and PR curve motivates to generate reference image using consensus of different edge detectors. The minimean and minimax methods are considered for automatic generation of ground truth edge map using optimum vote. The experimental results on various images containing a wide range of natural images of varying illumination, contrast and textural details yield the potential of Canny edge detector over other methods.

Due to the ever increasing demand for high speed and time critical tasks in many image processing applications, an efficient hardware architecture for Canny edge detector is implemented in VHDL. The studied implementation technique adopts parallel architecture of Field Programmable Gate Array (FPGA) to accelerate the process of edge detection via. Canny's algorithm. We have simulated the considered architecture in Modelsim 10.4a student edition to demonstrate the potential of parallel processing for edge detection. The analysis and implementation technique may encourage and serve as a

basis building block for several complex computer vision applications.

7.2 Future work

The performance of studied algorithm for automatic reference edge map generation using consensus of different edge detector may be improved by including more edge detection algorithms. Moreover the process of parallel operation for Canny edge detector may be accelerated. The choice of different edge detection algorithms and the hardware architecture for its implementation is left as future scope of this thesis.

Bibliography

[1] M. Sezgin and B. Sankur, "Selection of thresholding methods for nondestructive testing applications," in *Image Processing, 2001. Proceedings. 2001 International Conference on*, vol. 3. IEEE, 2001, pp. 764–767.

[2] N. L. Fernández-García, A. Carmona-Poyato, R. Medina-Carnicer, and F. J. Madrid-Cuevas, "Automatic generation of consensus ground truth for the comparison of edge detection techniques," *Image and Vision Computing*, vol. 26, no. 4, pp. 496–511, 2008.

[3] S. M, "Development of an fpga based image processing intellectual property core," M.Tech dissertation, NIT Rourkela, May 2014.

[4] D. Ziou, S. Tabbone *et al.*, "Edge detection techniques-an overview," *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, vol. 8, pp. 537–559, 1998.

[5] R. C. Gonzalez and R. E. Woods, "Digital image processing," 2002.

[6] L. G. Roberts, "Machine perception of three-dimensional soups," Ph.D. dissertation, Massachusetts Institute of Technology, 1963.

[7] M. A. Oskoei and H. Hu, "A survey on edge detection methods," *University of Essex, UK*, 2010.

[8] R. O. Duda, P. E. Hart *et al.*, *Pattern classification and scene analysis*. Wiley New York, 1973, vol. 3.

[9] M. C. Morrone, J. Ross, D. C. Burr, and R. Owens, "Mach bands are phase dependent," *Nature*, vol. 324, no. 6094, pp. 250–253, 1986.

[10] R. Owens, S. Venkatesh, and J. Ross, "Edge detection is a projection," *Pattern Recognition Letters*, vol. 9, no. 4, pp. 233–244, 1989.

[11] P. Kovesi, "Image features from phase congruency," *Videre: Journal of computer vision research*, vol. 1, no. 3, pp. 1–26, 1999.

[12] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 207, no. 1167, pp. 187–217, 1980.

[13] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 679–698, 1986.

[14] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets," *PloS one*, vol. 10, no. 3, p. e0118432, 2015.

[15] D. J. Bryant and D. W. Bouldin, "Evaluation of edge operators using relative and absolute grading," in *Conference on Pattern Recognition and Image Processing*, vol. 1, 1979, pp. 138–145.

[16] K. Bowyer, C. Kranenburg, and S. Dougherty, "Edge detector evaluation using empirical roc curves," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1. IEEE, 1999.

[17] Y. Yitzhaky and E. Peli, "A method for objective edge detection evaluation and detector parameter selection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 8, pp. 1027–1033, 2003.

[18] A. J. Baddeley, "An error metric for binary images," *Robust computer vision*, vol. 5978, 1992.

[19] L. Ding and A. Goshtasby, "On the canny edge detector," *Pattern Recognition*, vol. 34, no. 3, pp. 721–725, 2001.

[20] D. G. Bailey, *Design for embedded image processing on FPGAs*. John Wiley & Sons, 2011.

[21] P. J. Ashenden, "Digital design. an embedded system approach using vhdl," 2008.

Authors Biography

Nidhi Panda was born to Mr. Devendra Panda and Mrs. Madhu Panda on 15th February, 1992 at Raigarh, Chhattisgarh, India. She obtained her Bachelors degree in Electronics and Telecommunication Engineering from Disha Institute of Management and Technology, Raipur, Chhattisgarh in 2013. She joined the Department of Electrical Engineering, National Institute of Technology, Rourkela in July 2014 as an Institute Scholar to pursue Master of Technology.